1. Order the following list of functions by their big-Oh notation. Simplify the function notation using basic rules for logarithms and exponents in order to make their relative complexity evident.

$6n \log n$	2^{100}	$\log \log n$	$\log^2 n$	$2^{\log n}$
2^{2n}	$5n$	$n^{0.01}$	$\frac{1}{n}$	2^n
$3n^{0.5}$	$4^{\log n}$	$n^2 \log n$	$\sqrt{\log n}$	$n \ \log_4 n$
4^n				

Function	Simplified function	
		Slowest growing
		Fastest growing

2. Suppose we perform a sequence of operations on a queue data structure. After every n operations we make a copy of the entire queue for debugging purposes. Show that the cost of n operations (including the copy) is O(n) using the accounting method.

3. Suppose we have implemented a k-bit counter with a k-element binary array. The counter is initially 0. The only available operation is increment(A) which adds 1 to the current number.

- What is the worst-case running time of increment?
- What is the worst-case complexity for a sequence of k-increment?
- Use the potential method to find a better estimate.

4. Suppose we have 20 singleton sets, numbered 0 through 19, and we call the operation union(find(i),fin(i+5)), for i = 0,1,2,…..,14.
Draw a picture of the tree-based representation of the sets that result, assuming we don’t implement the union-by-size and path compression techniques.

5. Repeat exercise (4) assuming that we now implement both techniques